Modeling and Decoding Motor Cortical Activity Using a Switching Kalman Filter

Modeling and Decoding Motor Cortical Activity Using a Switching Kalman Filter

Wei Wu*, Michael J. Black, *Member, IEEE*, David Mumford, *Member, IEEE*, Yun Gao, Elie Bienenstock, and John P. Donoghue, *Member, IEEE*

***Abstract*—We present a switching Kalman filter model for the real-time inference of hand kinematics from a population of motor cortical neurons. Firing rates are modeled as a Gaussian mixture where the mean of each Gaussian component is a linear function of hand kinematics. A "hidden state" models the probability of each mixture component and evolves over time in a Markov chain. The model generalizes previous encoding and decoding methods, addresses the non-Gaussian nature of firing rates, and can cope with crudely sorted neural data common in on-line prosthetic applications.**

***Index Terms*—Mixture model, motor cortex, neural decoding, neural prosthesis, switching Kalman filter.**

I. Introduction

RECENT results have demonstrated the feasibility of continuous neural control of devices such as computer cursors using implanted electrodes [13], [28], [29], [33]. These results are enabled by a variety of mathematical "decoding" methods that produce an estimate of the system "state" (e.g., hand position) from a sequence of measurements (e.g., the firing rates of a population of cells). Here, we present a new model based on the switching Kalman filter [19] that generalizes previous approaches, achieves real-time decoding, and is appropriate for continuous neural-prosthetic control tasks.

We model the probability of the firing rates of the population at an instant in time as a Gaussian mixture where the mean of each Gaussian is a linear function of the hand kinematics. This mixture contains a "hidden state," or weight, that assigns a probability to each linear Gaussian term in the mixture. The evolution of this hidden state over time is modeled as a Markov chain. The expectation-maximization (EM) algorithm is used to fit this mixture model to training data that consists of measured hand kinematics (position, velocity, acceleration) and the firing rates of a small population of cells recorded with a chronically implanted multi-electrode array. Decoding of neural test data is achieved using the switching Kalman filter (SKF) algorithm [19]. Quantitative results show that the SKF outperforms the traditional linear Gaussian Kalman filter in the decoding of hand movement. These results suggest that the SKF provides a real-time decoding algorithm that may be appropriate for neural prosthesis applications.

The approach addresses a number of key issues and by doing so improves decoding accuracy over previous methods. First, the mixture model represents non-Gaussian distributions of firing rates. Previously, particle filtering has been proposed for modeling and decoding arbitrary, non-Gaussian, neural activity [4]. While general, this approach is computationally expensive and currently inappropriate for real-time decoding. The SKF can model non-Gaussian activity while maintaining many of the computational advantages of traditional linear Gaussian models; this is critical for neural prostheses. The SKF approach also addresses a common problem with on-line neural data. In prosthetic applications, individual electrodes may pick up activity of multiple cells and on-line spike detection and sorting techniques must be employed. These techniques tend to be based on simple thresholds and waveform analysis (e.g., [20]) and may result in multiple units being classified as a single cell [34]. In this respect, prosthetic applications differ somewhat from work on off-line encoding/decoding where careful spike sorting may be possible.

Our focus is on the real-time decoding of a continuous movement signal from population activity in the arm area of primary motor cortex (MI); see [25] and [27] for brief overviews. Roughly, the primary methods for decoding MI activity include the population vector algorithm [9], [13], [17], [18], [24], [29], linear filtering [21], [28], [33], artificial neural networks [22], [33], and probabilistic methods [4], [26], [35], [36]. The majority of these approaches model a linear relationship between the firing rates of motor cortical neurons and hand kinematics.

The population vector approach is the oldest method and was pioneered by Georgopoulos and colleagues in early 1980s [7], [9] to model the encoding of hand movement direction by a population of MI neurons. This work has led to a number of further observations that show that motor cortical firing rates are related to hand position [15], velocity (movement direction and speed) [18], and acceleration [3]. The population vector algorithm has been used successfully to decode various hand movement tasks which include center-out reaching [18], sinusoid tracing [23], spiral tracing [17], and lemniscate tracing [24]. Recently, the population vector algorithm was applied to the real-time neural control of three-dimensional (3-D) cursor movement in a center-out reaching task [29].

Manuscript received July 1, 2003; revised January 16, 2004. This work was supported in part by the Defence Advanced Research Projects Agency (DARPA) BioInfoMicro Program, by the NINDS Neural Prosthetics Program and Grant NS25074, and by the National Science Foundation (ITR Program Award 0113679). This work was previously presented at The 25th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Cancun, Mexico, Sept. 2003. *Asterisk indicates corresponding author.*

*W. Wu is with the Division of Applied Mathematics, Brown University, Providence, RI 02912 USA (e-mail: weiwu@dam.brown.edu).

M. J. Black is with the Department of Computer Science, Brown University, Providence, RI 02912 USA (e-mail: black@cs.brown.edu).

D. Mumford and Y. Gao are with the Division of Applied Mathematics, Brown University, Providence, RI 02912 USA (e-mail: david_mumford@brown.edu; gao@cfm.brown.edu).

E. Bienenstock is with the Division of Applied Mathematics and Department of Neuroscience, Brown University, Providence, RI 02912, USA (e-mail: elie@dam.brown.edu).

J. P. Donoghue is with the Department of Neuroscience, Brown University, Providence, RI 02912 USA (e-mail: john_donoghue@brown.edu).

Digital Object Identifier 10.1109/TBME.2004.826666

Although the population vector approach has shown success in these decoding and neural control tasks, it makes a number of assumptions [8] which are difficult to achieve in practical situations. For example, it assumes that the population uniformly represents the space of movement directions. With small populations, this is often not satisfied [12]. Additionally, the population vector method implicitly assumes a linear relationship between hand velocity and firing rate [36]. This has been shown many times to be a reasonable approximation when the data comes from well isolated cells. If activity from multiple units is combined by poor on-line spike sorting, this assumption may be violated.

The population vector method typically focuses on the relationship between hand velocity and firing rates. Continuous decoding of hand position then requires integration of velocity estimates over time. For general motions, errors quickly propagate reducing the effectiveness of the method. Consequently the approach is most often used for tasks with stereotyped motions such as reaching or for on-line control where the subject may adapt to compensate for deficiencies in the decoding method.

An alternative model uses linear regression [31] to compute fixed linear filters relating hand position to a vector of firing rates defined over a relatively long time window (typically 500 ms to 1.5 s) [21]. Due to its accuracy and efficiency, it has been successfully applied to real-time direct neural control tasks [28], [33].

Artificial neural networks have also been applied to neural decoding problems [11], [22]. They were shown to be successful in the real-time prediction of hand trajectory from neural ensembles in MI [33]. The results were not significantly different from that of the linear filter and, recently, an analysis of the representation learned by such networks reveals that they essentially encode a linear model (cosine tuning) as in the population vector method [22].

The artificial neural networks, as well as population vectors and linear filters, lack both a clear probabilistic model and a model of the temporal hand kinematics. Additionally, they provide no estimate of uncertainty and hence may be difficult to extend to the analysis of more complex temporal movement patterns. In contrast, a probabilistic formulation allows the mathematically principled combination of information and can take into account uncertainty in the data.

We have previously suggested the benefits of a probabilistic approach that uses data in small time windows (e.g., 50–100 ms or less) and integrates that information over time in a recursive fashion [4], [36]. In particular, we recently proposed a Kalman filter [35], [36], in which hand movement is encoded by a population of cells with a linear Gaussian model and is decoded using the Kalman filter algorithm. The method is based on an approximate generative model of neural firing. In particular, it assumes that the observed firing rates are a linear function of hand kinematics (position, velocity, and acceleration) and that they are corrupted by Gaussian noise.

This generative model is only a rough approximation and is violated in many ways. The distribution of firing rates is not Gaussian and the relationship to hand kinematics is only approximately linear. Moreover, poor spike sorting can compound these problems, resulting in decreased decoding performance for any linear approach. This can occur when a single electrode records the activity of multiple cells with similar waveforms that are difficult to distinguish.

We seek to systematically extend the Kalman filter encoding model to nonlinear relationships and non-Gaussian statistics and to evaluate the performance of such a model with respect to neural decoding. Unfortunately, fully general nonlinear models are difficult to learn from training data and the associated decoding methods are computationally expensive [4], [5].

Instead, the SKF model proposed here treats neural firing rates as a probabilistic *mixture* of linear models. If the encoding model of each cell is linear, then the SKF is better able to approximate the combination of firing rates. The method extends the Kalman model to exploit a mixture of linear Gaussian models. This mixture model, combined with a linear Gaussian model for the hand kinematics, is referred to as switching Kalman filter model (SKFM) [19]. It is also known as hybrid model, state-space model with switching, jump-linear system [10] and conditional dynamic linear model [2].

While such a model is more general than the simple linear Gaussian model, it still admits an efficient real-time decoding algorithm. We show how it generalizes to non-Gaussian firing models and how it can cope with poor spike sorting. In particular, we construct test data that is intentionally corrupted to simulate the effect of poor spike sorting. The firing rates of pairs of well isolated units are summed to produce synthetic mixed cells and we demonstrate how the SKFM is able to separate the combined activity to approximate the linear tuning properties of the individual units. The method is also tested on data recorded during a neural prosthetic control task. While on-line control is not tested here, the off-line reconstruction results demonstrate the appropriateness of the method for decoding and suggest that on-line experiments should be pursued.

II. Methods

A. Data Acquisition and Processing

To explore the SKF model and its application to neural decoding, we consider two datasets consisting of simultaneously recorded hand kinematics and neural activity. Both experiments used neural signals recorded with Bionic Technologies LLC (BTL) 100-electrode silicon arrays [16] which were chronically implanted in the arm area of primary motor cortex (MI) in macaque monkeys. Signals were amplified and sampled at 40 kHz/channel using a commercial recording system [14]. The experiments differ in the task being performed and the processing of the recorded waveforms.

The specific design of the task will effect the resulting encoding model. The common radial reaching tasks vary the direction of movement and consequently encoding models using this data focus on directional tuning. More general control tasks (e.g., computer cursor control) require full two-dimensional (2-D) control (at least). Consequently, we consider two tasks in which the hand motion spans a range of 2-D positions, velocities, and accelerations.

1) Pursuit Tracking Task: Electrophysiological recordings were made while the monkey performed a continuous tracking task [21]. The monkey viewed a computer monitor while gripping a two-link manipulandum that controlled the 2-D motion of a cursor on the monitor. In each trial, a target dot moved slowly and continuously on the monitor and the task required moving a feedback dot with the manipulandum so that it kept tracking the target within a given distance range. Hand positions were recorded every 50 ms and from this we computed velocity and acceleration using simple differencing.

A trial ended when the dot fell outside the tracking range. We eliminated the short trials (with duration less than 5 s). The majority of the remaining 182 trials were approximately 8–9 s

in duration. Note that the hand motions in this task were more general than those in the more common stereotyped tasks (e.g., "center-out" task in [29]) in that the motions spanned a range of directions, positions, and speeds (see [21] for details). They cannot, however, be considered natural motions.

During a trial, all neural activity that crossed a manually set threshold was digitized (12-bit voltage resolution) and saved to disk. Waveforms and their corresponding timestamps were saved for each electrode in the array. These waveforms were "sorted" by hand (off-line) using a commercial software [14]. Twenty five well-isolated units were detected and used for analysis. The empirical firing rate for each unit was calculated by counting the number of spikes within the previous 50-ms time window. The mean firing rate for each unit was then subtracted to obtain zero-mean data.

2) "Pinball" Task: In the "pinball" task [28], a target dot appeared at a random location on the monitor and the monkey was required to move the feedback dot with the manipulandum to hit the target. When the target was hit, it randomly jumped to a new 2-D location. In this task, the subject's hand motions were fast and unconstrained; this was more natural than the motion in the pursuit tracking task and simulated the motions needed to control a computer interface.

Each trial in this task was typically several minutes long. The data analyzed here includes two trials: one of approximately 3.5 min in length was used as training data and the other, approximately 1 min long, was used as test data for decoding.

As in the off-line task, hand position was recorded and hand velocity and acceleration were computed from the hand positions. Here, however, the task was designed for on-line neural control in [28] and, consequently, the spike acquisition exploited simple thresholding of waveforms rather than manual, off-line, sorting. As a result, the activity recorded with a single electrode may be the combination of the spiking activity of multiple cells. Data was recorded from 42 channels, action potentials crossing manually set thresholds were detected, and the firing rate for each channel was computed in 70 ms time bins. The mean firing rates in this task were larger than that in the pursuit tracking task due to the more rapid motions and the possible combination of units. The distribution of firing rates was also less Gaussian so we applied a square-root transform to the firing data as suggested in [18]. The mean firing rate for each unit was then subtracted to obtain zero-mean data.

3) Data Preprocessing: In the work that follows, we fit a Gaussian mixture model to the firing data in which each component of the model had a full covariance matrix (i.e. 42×42). Given the large number of units, correlations between their firing activity, and a limited amount of training data, fitting multiple covariance matrices can be computationally infeasible. To deal with this issue, we reduced the dimensionality of the input firing rates using principal component analysis (PCA) [1].

Let $\mathbf{z}_t = [z_{i,t}, \ldots, z_{n,t}]^T \in \mathbb{R}^n$ represent the zero-mean firing rates of n cells at time bin t. We constructed a matrix $A = [\mathbf{z}_1 \ldots \mathbf{z}_M]$ for the M time bins in the training data. Since the mean firing rate for each cell is zero, the covariance matrix Σ of the firing rate vectors is given by $\Sigma = AA^T$. Since Σ is symmetric and positive semi-definite, there exists an orthonormal matrix U and a nonnegative diagonal matrix Λ such that $\Sigma = U\Lambda U^T$. Each column in U is an eigenvector and each diagonal entry in Λ is an eigenvalue which reveals the variance in the corresponding eigenvector direction. The total variance is the sum of all these eigenvalues. If the firing rates of cells covary then the covariance matrix will not be full rank and some of the eigenvalues will be small. In this case, fitting the covariance matrices in the next section will be ill-conditioned.

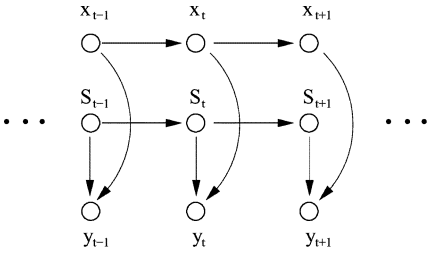


Fig. 1. Graphical model representation for SKFM: it is a combination of a state-space model and hidden Markov model. Both states $\mathbf{x}_t$ and switching labels S_t are assumed to be Markov over time, and conditioned on the states and labels, the observation likelihood is a linear Gaussian model.

To cope with this problem, let $\tilde{U}$ be a submatrix of U which only contains those eigenvectors with correspondingly large eigenvalues. Let $\tilde{Z} = \tilde{U}^T A$, be the projection of the original firing rate vectors on the lower dimensional subspace spanned by the columns of $\tilde{U}$. In the pinball task data, the projection of the 42 firing rates onto a 39-dimensional subspace resulted in a loss of less than 1% of the total variance. Rather than work with the original firing rates we work with their projections and refer to each column in $\tilde{Z}$ as a vector of firing rates for simplicity.

B. Computational Methods

In the SKFM, the hand movement (position, velocity and acceleration) is modeled as the system *state* and the neural firing rate is modeled as the *observation* (measurement). Let the *state* of the hand at the current instant in time be $\mathbf{x}_t = [x, y, v_x, v_y, a_x, a_y]_t^T \in \mathbb{R}^6$, which represents x-position, y-position, x-velocity, y-velocity, x-acceleration, and y-acceleration at time $t\Delta t$, where $\Delta t = 70$ ms (pinball task) or 50 ms (off-line task) in our experiments. The observations $\mathbf{y}_t \in \mathbb{R}^K$ represent a $K \times 1$ vector containing the firing rates at time t for the K observed neurons within Δt.

Our goal is to model the probabilistic relationship between $\mathbf{x}_t$ and $\mathbf{y}_t$. To that end, we exploit the SKF model illustrated in Fig. 1 [19]. In contrast to the standard Kalman filter [32], the SKF introduces a discrete "switching" variable S_t. The intuition is that the observations may be generated by different models represented by this switching variable.

Specifically, we model the joint probability distribution over states ($\{\mathbf{x}_t\}$), observations ($\{\mathbf{y}_t\}$), and switching variables ($\{S_t\}$) is

$$p(\{\mathbf{x}_t, \mathbf{y}_t, S_t\}) = \left[p(S_1)\prod_{t=2}^{T} p(S_t|S_{t-1})\right] \times \left[p(\mathbf{x}_1)\prod_{t=2}^{T} p(\mathbf{x}_t|\mathbf{x}_{t-1})\right]\left[\prod_{t=1}^{T} p(\mathbf{y}_t|\mathbf{x}_t, S_t)\right] \quad (1)$$

where $t = 1, 2, \ldots, T$ represent discrete time bins. Conditioned on the hidden switching variable, the probability of observing the firing rate vector is given by

$$p(\mathbf{y}_t|\mathbf{x}_t) = \sum_{j=1}^{N} p(S_t = j)p(\mathbf{y}_t|\mathbf{x}_t, S_t = j). \quad (2)$$

This simply states that the probability of observing the firing rates $\mathbf{y}_t$ conditioned on the kinematics $\mathbf{x}_t$ is represented by a mixture model. The term $p(S_t = j)$ represents the probability

of being in model j at time t for each of the $j = 1 \ldots N$ possible models.

The term $p(\mathbf{y}_t|\mathbf{x}_t, S_t = j)$ represents the likelihood of observing the firing rates conditioned on both the kinematics and the particular model j. We assume that each model in the mixture is linear and Gaussian. Consequently, we write

$$p(\mathbf{y}_t|\mathbf{x}_t, S_t = j) = \mathcal{N}(\mathbf{H}_j\mathbf{x}_t, \mathbf{Q}_j) \tag{3}$$

where $\mathcal{N}(\mathbf{H}_j\mathbf{x}_t, \mathbf{Q}_j)$ denotes a Gaussian distribution with mean $\mathbf{H}_j\mathbf{x}_t$. The matrix $\mathbf{H}_j \in \mathbb{R}^{K\times 6}$ linearly relates the hands state to the neural firing. The noise covariance matrix is $\mathbf{Q}_j \in \mathbb{R}^{K\times K}$. In the case of the standard Kalman filter, $N = 1$ and consequently there is only a single linear model of this form.

Note that the physical relationship between the firing rate and hand kinematics means there exists a time lag between them [18], [21]. In previous work, we noted that the optimal lag was approximately 140 ms in the pinball task [36] and 150 ms in the pursuit tracking task [35]. We used the same lags here for both encoding and decoding and the likelihood term should be written $p(\mathbf{y}_{t-lag}|\mathbf{x}_t, S_t = j)$ but to simplify the notation, we omit the lag time in the equations.

We assume the hidden states $S_1, S_2, \cdots, S_T$ form a first-order Markov chain as illustrated in Fig. 1; that is

$$p(S_t = j) = \sum_{i=1}^{N} p(S_t = j|S_{t-1} = i)p(S_{t-1} = i) \tag{4}$$

where we denote

$$c_{ij} = p(S_t = j|S_{t-1} = i), \quad 1 \le i, j \le N. \tag{5}$$

We represent these state transition probabilities as a *transition matrix* $\mathbf{C} = \{c_{ij}\}$.

The kinematic state is also assumed to form a Markov chain represented by the system model

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{A}\mathbf{x}_{t-1}, \mathbf{W}) \tag{6}$$

where $\mathbf{A} \in \mathbb{R}^{6\times 6}$ is the coefficient matrix linearly relating hand kinematics at time $t-1$ to the kinematics at time t. The noise covariance matrix is $\mathbf{W} \in \mathbb{R}^{6\times 6}$ represents the uncertainty in this prediction. Note that this system model is identical to that in the traditional Kalman filter.

Encoding: In practice, we need to estimate all the parameters $\mathbf{A}$, $\mathbf{W}$, $\mathbf{H}_{1:N}$, $\mathbf{Q}_{1:N}$, $\mathbf{C}$ from training data, in which both hand kinematics $\{\mathbf{x}_t\}$ and firing rates $\{\mathbf{y}_t\}$ are known, but the switching labels $\{S_t\}$ are hidden. Therefore, we estimate all the parameters by maximizing likelihood $p(\{\mathbf{x}_t, \mathbf{y}_t\})$

$$\begin{aligned}&\operatorname{argmax}_{A,W,H_{1:N},Q_{1:N},C}\, p(\{\mathbf{x}_t, \mathbf{y}_t\})\\ &= \operatorname{argmax}_{A,W,H_{1:N},Q_{1:N},C}\, p(\{\mathbf{x}_t\})\, p(\{\mathbf{y}_t\}|\{\mathbf{x}_t\})\\ &= \operatorname{argmax}_{A,W}\, p(\{\mathbf{x}_t\}) \operatorname{argmax}_{H_{1:N},Q_{1:N},C}\, p(\{\mathbf{y}_t\}|\{\mathbf{x}_t\}).\end{aligned}$$

Using the linear Gaussian property of $p(\{\mathbf{x}_t\})$, we have

$$\operatorname{argmax}_{A,W}\, p(\{\mathbf{x}_t\}) = \operatorname{argmin}_{A,W} \sum_{t=2}^{T} \left[\log(\det \mathbf{W}) + (\mathbf{x}_t - \mathbf{A}\mathbf{x}_{t-1})^T\mathbf{W}^{-1}(\mathbf{x}_t - \mathbf{A}\mathbf{x}_{t-1})\right].$$

The above minimization has a closed-form solution as follows:

$$\mathbf{A} = \sum_{t=2}^{T} \mathbf{x}_t\mathbf{x}_{t-1}^T \left(\sum_{t=2}^{T} \mathbf{x}_{t-1}\mathbf{x}_{t-1}^T\right)^{-1}$$

$$\mathbf{W} = \frac{1}{T-1}\left(\sum_{t=2}^{T} \mathbf{x}_t\mathbf{x}_t^T - \mathbf{A}\sum_{t=2}^{T} \mathbf{x}_{t-1}\mathbf{x}_t^T\right).$$

The other term $p(\{\mathbf{y}_t\}|\{\mathbf{x}_t\}) = \sum_{\{S_t\}} p(\{\mathbf{y}_t, S_t\}|\{\mathbf{x}_t\})$ contains hidden variables $\{S_t\}$. While no closed form solution exists, the EM algorithm offers an effective way to estimate all the parameters. Denoting $\theta = (\mathbf{H}_{1:N}, \mathbf{Q}_{1:N}, \mathbf{C})$ and $p(\cdot|\cdots) = p(\cdot|\{\mathbf{x}_t, \mathbf{y}_t\}; \theta_k)$, we update θ_k to θ_{k+1} as

$$\theta_{k+1} = \operatorname{argmax}_\theta\, \mathbf{E}_{p(\{S_t\}|\ldots)} \log p(\{\mathbf{y}_t, S_t\}|\{\mathbf{x}_t\}; \theta).$$

The details of the maximization process can be found in [19]. We only show the updating result here

$$c_{ij} = \frac{\sum_{t=2}^{T} p(S_t = j, S_{t-1} = i|\cdots)}{\sum_{t=2}^{T} p(S_{t-1} = i|\cdots)}$$

$$\mathbf{H}_j = \left[\sum_{t=1}^{T} p(S_t = j|\cdots)\mathbf{y}_t\mathbf{x}_t^T\right]\left[\sum_{t=1}^{T} p(S_t = j|\cdots)\mathbf{x}_t\mathbf{x}_t^T\right]^{-1}$$

$$\mathbf{Q}_j = \frac{\sum_{t=1}^{T}\left[p(S_t = j|\cdots)\left(\mathbf{y}_t\mathbf{y}_t^T - \mathbf{H}_j\mathbf{x}_t\mathbf{y}_t^T\right)\right]}{\sum_{t=1}^{T} p(S_t = j|\cdots)}$$

where $i, j = 1, \ldots, N$ and the conditional probabilities of S_t, S_{t-1} can be calculated using standard dynamic programming techniques.

Decoding (Estimation): Given the probabilistic encoding model defined above, we turn to the problem of decoding; that is, reconstructing hand motion from the firing rates of the cells. Let $\mathbf{x}_{1:t}$ denote $\mathbf{x}_1, \cdots, \mathbf{x}_t$, and similarly for $\mathbf{y}_{1:t}$ and $S_{1:t}$. We seek the *a posteriori* mean $\hat{\mathbf{x}}_t = \mathrm{E}(\mathbf{x}_t|\mathbf{y}_{1:t})$ that minimizes the mean square error $\mathrm{E}((\mathbf{x}_t - \hat{\mathbf{x}}_t)^2|\mathbf{y}_{1:t})$. We achieve this using the efficient SKF algorithm which is briefly described here (see [19] for details).

Under the SKFM framework, the posterior distribution of the state is a mixture of Gaussians, but the number of mixture components grows exponentially with time; that is, assume the initial $p(\mathbf{x}_1|\mathbf{y}_1)$ is a mixture of N Gaussians (one for each value of S_1), then $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ is a mixture of N^t Gaussians (one for each possible sequence of $S_1, \cdots, S_t$). The SKF algorithm [19] approximates these N^t Gaussians with a mixture of N Gaussians at each time step t. The fixed number N over time is maintained by "collapsing" N Gaussians into one using moment matching, which can be shown to be the optimal approximation under the criterion of minimization of relative entropy between the Gaussians.

For neural prosthetic applications, the SKF algorithm is preferable to other sampling-based algorithms [2], [4] since, by "collapsing" the posterior at each time instant, it provides both a deterministic algorithm and a probabilistic model representing uncertainty in the estimate. The uncertainty may be important for later stages of analysis. We illustrate this uncertainty estimation in the following section on experimental data analysis.

The SKF decoding algorithm proceeds as follows. At time $t-1$, assume that the posterior distribution $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$ is approximated with a mixture of N Gaussians. That is

$$p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}) = \sum_{i=1}^{N} w_{t-1}^i p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}, S_{t-1} = i)$$

where the weight $w_{t-1}^i = p(S_{t-1} = i|\mathbf{y}_{1:t-1})$ and each component

$$p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}, S_{t-1} = i) \approx \mathcal{N}\left(\mathbf{x}_{t-1}^i, \mathbf{V}_{t-1}^i\right)$$

in which the mean and covariance of the system state are given by $\mathbf{x}_{t-1}^i = \mathrm{E}[\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}, S_{t-1} = i]$ and $\mathbf{V}_{t-1}^i = \mathrm{Cov}[\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}, S_{t-1} = i]$, $i = 1, \ldots, N$, respectively.

At time t, we will show how the posterior $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ is estimated as a Gaussian mixture and how the weight, mean and covariance of each Gaussian are obtained. At first, we expand the posterior as a linear sum

$$\begin{aligned} p(\mathbf{x}_t|\mathbf{y}_{1:t}) &= \sum_{j=1}^{N} w_t^j p(\mathbf{x}_t|\mathbf{y}_{1:t}, S_t = j) \\ &= \sum_{j=1}^{N} w_t^j \sum_{i=1}^{N} g_t^{ij} p(\mathbf{x}_t|\mathbf{y}_{1:t}, S_t = j, S_{t-1} = i) \\ &= \sum_{i,j=1}^{N} w_t^{ij} p(\mathbf{x}_t|\mathbf{y}_{1:t}, S_t = j, S_{t-1} = i) \end{aligned}$$

where the weight $w_t^j = p(S_t = j|\mathbf{y}_{1:t})$, $g_t^{ij} = p(S_{t-1} = i|\mathbf{y}_{1:t}, S_t = j)$, and $w_t^{ij} = w_t^j g_t^{ij} = p(S_t = j, S_{t-1} = i|\mathbf{y}_{1:t})$, $i, j = 1, \cdots, N$.

For each i and j, $p(\mathbf{x}_t|\mathbf{y}_{1:t}, S_t = j, S_{t-1} = i)$ is a Gaussian density function which can be calculated using the standard Kalman filter subroutine **filter** (shown in the Appendix). Since the Kalman filter produces a Gaussian estimate of the state, we have

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}, S_t = j, S_{t-1} = i) = \mathcal{N}\left(\mathbf{x}_t^{ij}, \mathbf{V}_t^{ij}\right)$$

where the mean $\mathbf{x}_t^{ij} = \mathrm{E}[\mathbf{x}_t|\mathbf{y}_{1:t}, S_t = j, S_{t-1} = i]$ and the covariance matrix $\mathbf{V}_t^{ij} = \mathrm{Cov}[\mathbf{x}_t|\mathbf{y}_{1:t}, S_t = j, S_{t-1} = i]$.

Let $l_t^{ij} = p(\mathbf{y}_t|\mathbf{y}_{1:t-1}, S_t = j, S_{t-1} = i)$ be the likelihood of the observing the firing rate $\mathbf{y}_t$. This can be calculated as a by-product of the Kalman filter algorithm. Using the above notation, we obtain an expressions for the following weights at time t: $w_t^{ij} = l_t^{ij} c_{ij} w_{t-1}^i / \sum_{ij} l_t^{ij} c_{ij} w_{t-1}^i$, $w_t^j = \sum_i w_t^{ij}$, and $g_t^{ij} = w_t^{ij}/w_t^j$.

Following the above analysis, we see that $p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \sum_{i,j=1}^{N} w_t^{ij} \mathcal{N}(\mathbf{x}_t^{ij}, \mathbf{V}_t^{ij})$ is a mixture of N^2 Gaussians. To reduce the number of mixture components to N, we combine the N Gaussians that have same label at time t to form a single Gaussian. That is, for each j, $p(\mathbf{x}_t|\mathbf{y}_{1:t}, S_t = j) = \sum_i g_t^{ij} \mathcal{N}(\mathbf{x}_t^{ij}, \mathbf{V}_t^{ij})$ is a mixture with N components; we approximate this as a single Gaussian by matching the mean and covariance matrix (using the subroutine **collapse** in the Appendix) that is

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}, S_t = j) \approx \mathcal{N}\left(\mathbf{x}_t^j, V_t^j\right)$$

where the mean $\mathbf{x}_t^j = \mathrm{E}[\mathbf{x}_t|\mathbf{y}_{1:t}, S_t = j]$, and the covariance matrix $\mathbf{V}_t^j = \mathrm{Cov}[\mathbf{x}_t|\mathbf{y}_{1:t}, S_t = j]$, $j = 1, \cdots, N$. Finally, we have the desired mixture approximation to the posterior probability of the system state conditioned on the measurements

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) \approx \sum_{j=1}^{N} w_t^j \mathcal{N}\left(\mathbf{x}_t^j, V_t^j\right).$$

In summary, the following algorithm shows how the posterior distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ is approximated by a mixture of N Gaussians $\sum_j w_t^j \mathcal{N}(\mathbf{x}_t^j, \mathbf{V}_t^j)$ at time step t.

From time step $t-1$ to t

$$\left[\mathbf{x}_t^{ij}, \mathbf{V}_t^{ij}, l_t^{ij}\right] = \mathbf{filter}\left(\mathbf{x}_{t-1}^i, \mathbf{V}_{t-1}^i, \mathbf{y}_t, \mathbf{H}_j, \mathbf{Q}_j, \mathbf{A}, \mathbf{W}\right)$$

$$w_t^{ij} = \frac{l_t^{ij} c_{ij} w_{t-1}^i}{\sum_{ij} l_t^{ij} c_{ij} w_{t-1}^i}$$

$$w_t^j = \sum_i w_t^{ij}$$

$$g_t^{ij} = \frac{w_t^{ij}}{w_t^j}$$

$$\left[\mathbf{x}_t^j, \mathbf{V}_t^j\right] = \mathbf{collapse}\left(\left\{\mathbf{x}_t^{ij}, \mathbf{V}_t^{ij}, g_t^{ij}\right\}_i\right).$$

When a single estimate and its uncertainty is desired, these can be computed from the mixture model. The state estimate $\hat{\mathbf{x}}_t$ and its error covariance $\hat{\mathbf{V}}_t$ (obtained by the same approach as in **collapse**) are given by

$$\hat{\mathbf{x}}_t = \sum_j w_t^j \mathbf{x}_t^j,$$

$$\hat{\mathbf{V}}_t = \sum_j w_t^j \left(\mathbf{V}_t^j + \left(\mathbf{x}_t^j - \hat{\mathbf{x}}_t\right)\left(\mathbf{x}_t^j - \hat{\mathbf{x}}_t\right)^T\right).$$

III. Results

We summarize three sets of experiments. The first involves the data from the pursuit tracking task in which the electrophysiological recordings were carefully sorted off-line. From this set, we constructed a second, mixed, data set that simulated errors in spike sorting. Finally, we considered data from an on-line scenario where spike detection was performed in real-time; we expect the quality of the spike sorting to be poorer than in the pursuit tracking task resulting in mixed units.

A. Pursuit Tracking Task

Evaluation of the encoding and decoding performance was performed using cross-validation. We evenly partitioned the 182 trials into seven segments (i.e., each segment had 26 trials). Then for each experiment, 156 trials (six segments) were used as training data to learn the probabilistic model, and the remaining 26 trials (one segment) were used to test the decoding accuracy. The SKFM was trained with two mixture components ($N = 2$). This training/testing process was performed 7 times and each trial was used in the decoding stage exactly once. Despite the large amount of training data, computing the encoding model took only a few minutes. Decoding was performed in real time.

Since each testing trial was very short, we let the initial hand kinematics (at time 0) equal the true known initial condition. The SKF was then applied to reconstruct the hand trajectory; a few example reconstructions are shown in Fig. 2.

Errors are reported in terms of mean squared reconstruction error (MSE) and the correlation coefficient (CC) for x, y position. Assume $(\hat{x}_t, \hat{y}_t)$ is the estimate for the true position (x_t, y_t), $t = 1, \cdots, T$, then MSE and CC are defined as follows:

$$\mathrm{MSE} = \frac{1}{T}\sum_{t=1}^{T}\left((x_t - \hat{x}_t)^2 + (y_t - \hat{y}_t)^2\right)$$

$$CC = \left(\frac{\sum_t (x_t - \bar{x})(\hat{x}_t - \bar{\hat{x}})}{\sqrt{\sum_t (x_t - \bar{x})^2 \sum_t (\hat{x}_t - \bar{\hat{x}})^2}}, \frac{\sum_t (y_t - \bar{y})(\hat{y}_t - \bar{\hat{y}})}{\sqrt{\sum_t (y_t - \bar{y})^2 \sum_t (\hat{y}_t - \bar{\hat{y}})^2}}\right).$$

For prosthetic applications, MSE may be particularly relevant since it measures the positional accuracy of the neural reconstruction.

The waveforms for this dataset were carefully sorted off-line reducing (but not eliminating) the chance of multiple units being treated as one. The neural activity for this dataset has previously

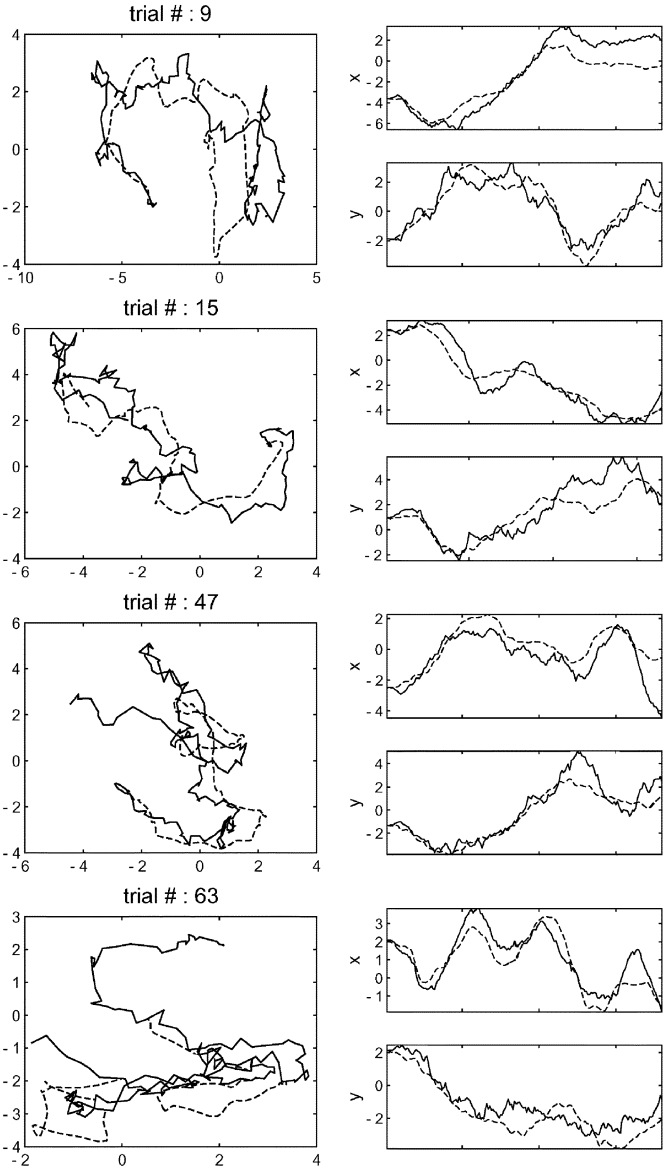


Fig. 2. Reconstruction on four test trials (each row is one): true hand trajectory (dashed) and reconstruction using the SKF (solid). Left column: the trajectories in 2-D. Right column: the trajectories by their x and y components.

been shown to be approximately linearly related to hand kinematics [35]. Consequently, we expected the SKF and Kalman filter to have similar performance.

The comparison of decoding accuracy for the SKF and Kalman filter is shown in Table I. The table shows the percentage of the time that the SKF was superior to (had lower reconstruction error than) the Kalman filter. The first column of results [labeled "25 (25S, 0D)"] shows that the SKF and Kalman filter had roughly the same decoding performance on this well sorted data set. The differences between the models were not statistically significant.

B. Mixed Cells

In practical neural prosthetic applications, real-time computational constraints may mean that spiking activity is poorly sorted. To test the ability of the SKF to cope with such data, we constructed artificial spike trains by randomly selecting pairs of cells from the pursuit tracking dataset and combining their responses. The resulting firing rate for each "double" cell was the sum of the individual firing rates. This synthetic set of cells simulated what would happen if two units were poorly separated and their waveforms were judged as coming from the same cell.

In particular, we tested three different situations with different combinations of the original 25 cells: 1) we randomly chose ten units from the 25, and summed their rate functions to form five new "units" containing spikes from pairs of original cells; 2) we combined 20 cells to form ten doubles; and 3) we combined 24 cells to form 12 doubles. Once again, the SKFM was trained with two mixture components.

We tested the performance of SKF on this simulated experimental data and compared it with Kalman filter. The fourth to sixth column of Table I, show that SKF outperforms the KF in terms of reconstruction accuracy and the results are statistically significant (the p-values using a sign test are all less than 5%).

Fig. 3 provides some intuition regarding the performance of the SKFM on the double data. The figure shows the empirical firing rates for a selection of cells in terms of position or velocity. Note that dark blue corresponds to no measurement while brighter colors of red indicate high firing rate. When the a linear Gaussian model was used to fit this data, the resulting encodings were linear (planar) as expected. The figure also shows some of the double "units" that combine the activity of two cells. In this case, the simple linear fit did not well model either of the original cells. The SKFM however fit multiple linear models (two here) and was able to approximate the original linear functions in these cases.

C. Pinball Task

We also tested the SKF model and its decoding algorithm on the data from the pinball task. For this on-line task, the spike sorting was less accurate and we hypothesized that the SKF would be more appropriate. Experimentally, we found that approximately 3.5 min of training data sufficed for accurate reconstruction (this is similar to the result for fixed linear filters reported in [28]). Learning the model took approximately 1 min on a Pentium-III 866.

At the beginning of the test trial, we assumed that we had no information about the hand kinematics and, consequently, let the predicted initial condition equal the average hand kinematics in training data. Both the SKF and Kalman filter were used to decode the test data and the results are reported in Table II. The results indicated that, the SKF gave a slightly more accurate reconstruction than the Kalman filter. Most critically, it gave an 8% reduction in MSE.

Fig. 4 shows the SKF reconstruction of the first 20 s of test data (distinct from the training data) for each component of the state variable (position, velocity and acceleration in x and y). The reconstructed trajectories are smooth and visually similar to the true ones (i.e., high Correlation coefficient). Note that the ground-truth velocity and acceleration curves are computed from the position data with simple differencing. As a result these plots are quite noisy making an evaluation of the reconstruction difficult.

For the SKF, the posterior distribution of the state is assumed to be a mixture of Gaussians. The hand state and its uncertainty are estimated as the mean and covariance matrix of the mixture. The 95% confidence interval of this estimation is shown for both x and y-position in Fig. 5 (calculated as the $[\mu - 2\sigma, \mu + 2\sigma]$, where μ, σ are the mean and standard deviation in the x or y component). From Fig. 5, we observe that the true positions were typically within this confidence interval. Such a

TABLE I

COMPARISON OF THE SKF AND KF. NOTATION: FOR EXAMPLE, IN THE LAST COLUMN, "13 (1S, 12D)" DENOTES THE USE OF 13 NEW "UNITS" ONE 1 IS SINGLE (1S) FROM THE ORIGINAL DATA AND 12 ARE DOUBLE (12D). THE DOUBLE "UNITS" ARE THE RESULT OF COMBINING PAIRS OF SINGLE UNITS (SEE TEXT). RESULTS: FOR EXAMPLE, 66.48% DENOTES THAT THE SKF HAS LOWER MSE IN 66.48% OF THE 182 TRIALS. 1.22e-5 IS THE p-VALUE WHICH SHOWS THE SIGNIFICANCE OF THE RESULT. THE p-VALUE MEANS THAT IF WE ASSUME THE KF AND SKF HAVE THE SAME DECODING ACCURACY, THEN THE PROBABILITY OF OBSERVING LOWER ERROR FROM THE SKF 66.48% OF THE TIME IS ONLY 1.22e-5. THE ASSUMPTION CAN THEN BE REJECTED. HERE THE p-VALUES ARE CALCULATED BY THE SIGN TEST [30]

	# of units	25 (25S, 0D)	20 (15S, 5D)	15 (5S, 10D)	13 (1S, 12D)
Mean Squared Error (MSE)	superiority	53.85%	61.54%	62.09%	66.48%
	p-value	3.35e-1	2.40e-3	1.40e-3	1.22e-5
Correlation Coefficient (CC)	superiority	48.35%	58.24%	57.69%	58.24%
	p-value	7.11e-1	3.16e-2	4.54e-2	3.16e-2

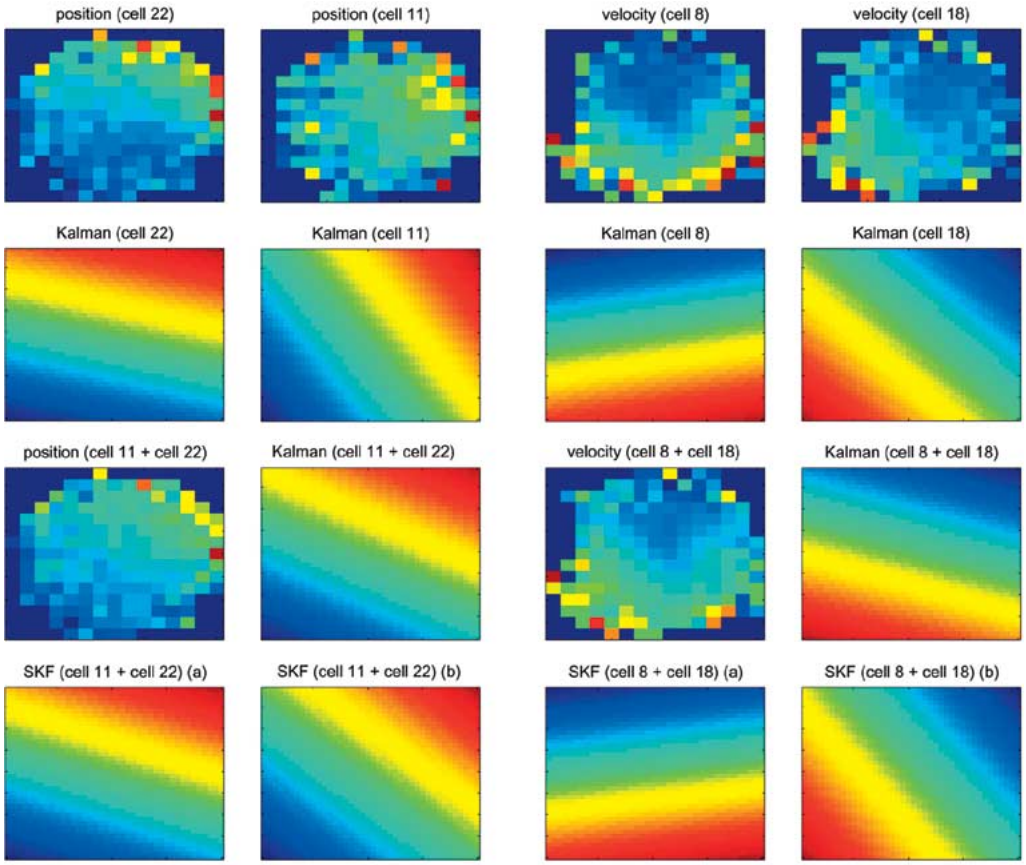


Fig. 3. Fitting multiple linear models: in the first two columns, the first row shows the empirical mean firing rate distributions of two cells with respect to hand position; the second row shows the corresponding linear fit to the data (Kalman filter model); the third row shows the empirical distribution of the sum of firing rates of these two cells and Kalman fit; the fourth row shows the two components recovered by the SKFM when applied the combined data. We see that these two linear components are very similar to the linear fits for the original two cells. The two right columns show the same behavior when we consider firing rate as a function of hand velocity. Note that the Kalman and SKF models used here fit linear models with respect to position, velocity and acceleration. The figure shows just the position or velocity component of these models.

TABLE II

RECONSTRUCTION ACCURACY USING THE KALMAN FILTER AND SKF

Method	Corr-Coef (x, y)	MSE (cm^2)
Kalman	(0.82, 0.93)	5.87
SKF	(0.84, 0.93)	5.39

confidence measure may be useful for later stages of processing that may exploit the neural control signal.

IV. DISCUSSION AND CONCLUSION

We have proposed an extension of the Kalman filter for neural decoding of motor cortical activity. The approach extends the observation model of the Kalman filter as a probabilistic mixture of linear Gaussian models. This mixture model is more general and can cope with non-Gaussian rate models and poor separation of units after spike sorting. These generalizations suggest that the approach may be more appropriate for neural control applications than previous linear methods such as the Kalman filter, linear filter, and population vector algorithm.

In particular, experiments that simulated the effect of poor spike sorting showed that the SKF performed significantly better than the traditional Kalman filter. This illustrates how the method can be applied when the assumptions of a simple linear Gaussian model are not met. To the extent that this simulation models real problems in on-line spike sorting, it suggests that that SKFM may be more appropriate and practical for neural prosthetic applications. Additionally, we found no performance

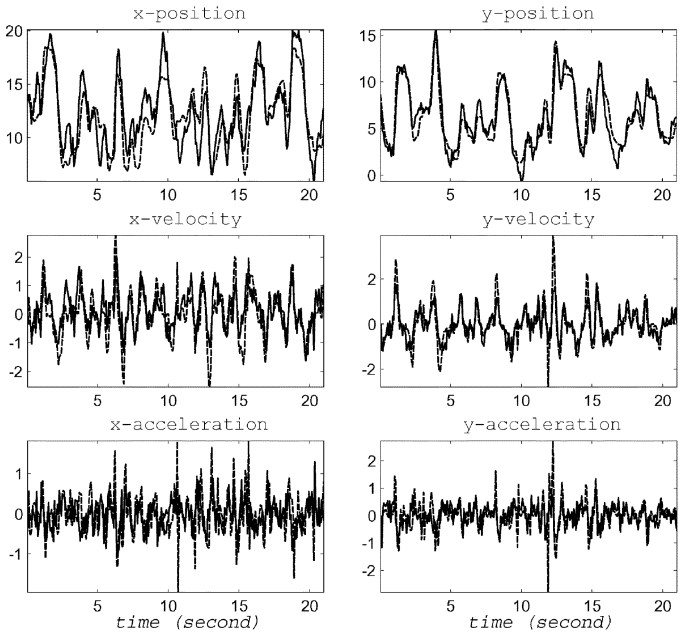


Fig. 4. Reconstruction of each component of the system state variable: true hand motion (dashed) and reconstruction using the SKF (solid). 20 s from a 1-min test sequence are shown.

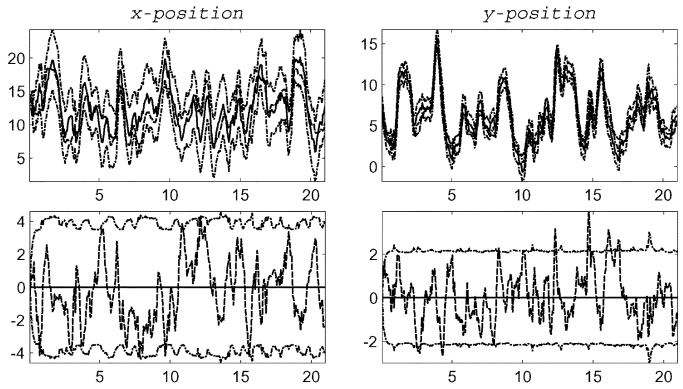


Fig. 5. Confidence estimation for the x and y-*position*: the first row shows the true (dashed), reconstructed trajectories (solid), and their 95% confidence range (dashdot). The second row is the normalized version obtained by subtracting the corresponding reconstruction; this shows the confidence intervals more clearly.

decrease with well sorted data. Consequently, the advantages of the approach may justify the increased complexity of the method relative to the simpler Kalman filter.

The mixture model can be efficiently learned with the EM algorithm using only few minutes of training data. The SKF algorithm provides an efficient decoding method that gives real-time estimates of hand position in short time windows (50–70 ms here) from the firing rates of a small population of cells. Relatively accurate decoding was achieve with populations of 25–42 cells. Reconstructions of 2-D hand motion were obtained for two different tasks and two different animals. The amount of data available for the Pinball task was limited thus making it impossible to state the statistical significance of the performance improvement in this case.

The hand motions explored here were more general than those in stereotyped tasks such as center-out reaching. Nevertheless, they were still simple compared with natural hand movements in daily life. In the future, we will explore more complex and natural movement and examine the extension of these methods to three dimensional reconstruction. For prosthetic applications, there is no reason to restrict the output of decoding to mimic natural hand motion; nonnatural mappings between neural activity and prosthetic control, however, are beyond the scope of this paper.

Here, we restricted our attention to off-line decoding accuracy. We conjecture that more accurate off-line algorithms should improve on-line control but this remains to be tested. In previous work we showed that the Kalman filter was more accurate at off-line decoding than linear filtering methods [36]. Here, we showed that the SKF is more accurate still. Our current work is exploring on-line neural cursor control and will compare the SKF with the Kalman filter and linear regression methods.

It is worth noting that reasonably accurate reconstructions of hand trajectories are possible from small populations of cells (e.g., 25–42 cells as considered here). In particular, data from the Pinball experiments was used for on-line control by Serruya *et al.* [28]. It remains an open question how the relative performance of different decoding methods will scale to larger populations.

In summary, the SKFM has many of the desirable properties of the Kalman filter (e.g., linear Gaussian models (when conditioned on the switching state), full covariance model, efficient encoding and decoding), while being more versatile and accurate. It can deal with violations (to some extent) of both the assumption of linearity and Gaussian noise. It also has an advantage over more general particle filter decoding methods [4] in that it is computationally efficient since it does not rely Monte Carlo sampling.

APPENDIX

A. Kalman Filter

$$[\mathbf{x}_t, \mathbf{V}_t, l] = \mathbf{filter}(\mathbf{x}_{t-1}, \mathbf{V}_{t-1}, \mathbf{y}_t, \mathbf{H}, \mathbf{Q}, \mathbf{A}, \mathbf{W})$$

We briefly summarize the discrete Kalman filter which is equivalent to the SKFM for $N = 1$ in (3) and (6). For details, see [32].

In the standard Kalman filter framework, the system $\mathbf{x}_t \in \mathbb{R}^n$ and measurement $\mathbf{y}_t \in \mathbb{R}^m$ are assumed to satisfy the following linear equations.

Measurement equation

$$\mathbf{y}_t = \mathbf{H}\mathbf{x}_t + \mathbf{q}_t. \tag{7}$$

System equation

$$\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t + \mathbf{w}_t \tag{8}$$

where $\mathbf{q}_t \sim \mathcal{N}(0, \mathbf{Q})$, $\mathbf{w}_t \sim \mathcal{N}(0, \mathbf{W})$, $\mathbf{H} \in \mathbb{R}^{m\times n}$, $\mathbf{Q} \in \mathbb{R}^{m\times m}$, $\mathbf{A} \in \mathbb{R}^{n\times n}$, and $\mathbf{W} \in \mathbb{R}^{n\times n}$, $t = 1, \cdots, T$.

The Kalman filter algorithm is a recursive estimation approach. At each time step t it has two steps: 1) prediction of the *a priori* state estimate $\hat{\mathbf{x}}_t^-$ and 2) updating this estimate with new measurement data to produce an *a posteriori* state estimate $\hat{\mathbf{x}}_t$. In particular, these steps are as follows.

I. **Time update equations**

At each time t, we obtain the *a priori* estimate from the previous time $t-1$, then compute its error covariance matrix $\mathbf{V}_t^-$ as follows:

$$\hat{\mathbf{x}}_t^- = \mathbf{A}\hat{\mathbf{x}}_{t-1} \tag{9}$$

$$\mathbf{V}_t^- = \mathbf{A}\mathbf{V}_{t-1}\mathbf{A}^T + \mathbf{W}. \tag{10}$$

II. **Measurement update equations**

Using the estimate $\hat{\mathbf{x}}_t^-$ and observation $\mathbf{y}_t$, we update the estimate using the measurement and compute the posterior error covariance matrix as follows:

$$\hat{\mathbf{x}}_t = \hat{\mathbf{x}}_t^- + \mathbf{K}_t\left(\mathbf{y}_t - \mathbf{H}\hat{\mathbf{x}}_t^-\right) \tag{11}$$

$$\mathbf{V}_t = (\mathbf{I} - \mathbf{K}_t\mathbf{H})\mathbf{P}_t^- \tag{12}$$

where $\mathbf{V}_t$ represents the state error covariance after taking into account the observation and $\mathbf{K}_t$ is the Kalman *gain* matrix given by

$$\mathbf{K}_t = \mathbf{V}_t^-\mathbf{H}^T\left(\mathbf{H}\mathbf{V}_t^-\mathbf{H}^T + \mathbf{Q}\right)^{-1}. \tag{13}$$

$\mathbf{K}_t$ produces a state estimate that minimizes the mean squared error of the estimation (see [6] for details). Note that $\mathbf{Q}$ is the measurement error matrix and, depending on the reliability of the data, the gain term $\mathbf{K}_t$ automatically adjusts the contribution of the new measurement to the state estimate.

The likelihood l is the probability of the observation $\mathbf{y}_t$ and can be calculated as a by-product of Kalman filter, i.e.,

$$l = \mathcal{N}\left(\mathbf{y}_t - \mathbf{H}\hat{\mathbf{x}}_t^-; 0, \mathbf{H}\mathbf{V}_t^-\mathbf{H}^T + \mathbf{Q}\right).$$

In the calling of this subroutine, the parameters $\mathbf{H}$, $\mathbf{Q}$, $\mathbf{A}$, and $\mathbf{W}$ have been estimated in the encoding stage and $\hat{\mathbf{x}}_{t-1}$, $\mathbf{V}_{t-1}$ are the estimated state and uncertainty (error covariance) at time $t-1$, so we can directly apply the above updating equations to propagate the state and uncertainty estimation from time step $t-1$ to t.

B. Collapsing a Mixture of Gaussians

$$[\mathbf{x}, \mathbf{V}] = \mathbf{collapse}\left(\{\mathbf{x}^i, \mathbf{V}^i, g^i\}_i\right)$$

The subroutine **collapse** approximates mixture of Gaussians as one Gaussian by moment matching. That is, the mean and covariance matrix of the one Gaussian are the same as that of the mixture. Therefore

$$\begin{aligned}\mathbf{x} &= \mathrm{E}\left(\mathbf{s}\sum_i g^i\mathcal{N}(\mathbf{s};\mathbf{x}^i,\mathbf{V}^i)\right)\\ &= \sum_i g^i\mathrm{E}\left(\mathbf{s}\mathcal{N}(\mathbf{s};\mathbf{x}^i,\mathbf{V}^i)\right)\\ &= \sum_i g^i\mathbf{x}^i\end{aligned}$$

and

$$\begin{aligned}\mathbf{V} &= \mathrm{E}\left((\mathbf{s}-\mathbf{x})(\mathbf{s}-\mathbf{x})^T\sum_i g^i\mathcal{N}(\mathbf{x}^i,\mathbf{V}^i)\right)\\ &= \sum_i g^i\mathrm{E}\left((\mathbf{s}-\mathbf{x})(\mathbf{s}-\mathbf{x})^T\mathcal{N}(\mathbf{x}^i,\mathbf{V}^i)\right)\\ &= \sum_i g^i\mathrm{E}\Big(\Big((\mathbf{s}-\mathbf{x}^i)(\mathbf{s}-\mathbf{x}^i)^T-(\mathbf{x}-\mathbf{x}^i)\mathbf{s}^T\\ &\qquad -\mathbf{s}(\mathbf{x}-\mathbf{x}^i)^T+\mathbf{x}\mathbf{x}^T+\mathbf{x}^i\mathbf{x}^{iT}\Big)\mathcal{N}(\mathbf{x}^i,\mathbf{V}^i)\Big)\\ &= \sum_i g^i\Big(\mathrm{E}\Big((\mathbf{s}-\mathbf{x}^i)(\mathbf{s}-\mathbf{x}^i)^T\mathcal{N}(\mathbf{x}^i,\mathbf{V}^i)\Big)\\ &\qquad +(\mathbf{x}^i-\mathbf{x})(\mathbf{x}^i-\mathbf{x})^T\Big)\\ &= \sum_i g^i\Big(\mathbf{V}^i+(\mathbf{x}^i-\mathbf{x})(\mathbf{x}^i-\mathbf{x})^T\Big).\end{aligned}$$

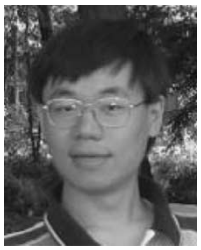
ACKNOWLEDGMENT

The authors thank M. Serruya, A. Shaikhouni, J. Dushanova, C. Vargas-Irwin, L. Lennox, and M. Fellows for their assistance.

REFERENCES

[1] C. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Oxford Univ. Press, 1995.
[2] R. Chen and J. S. Liu, "Mixture Kalman filter," *J. Roy. Stat. Soc.—Series B*, vol. 62, pp. 493–508, 2000.
[3] D. Flament and J. Hore, "Relations of motor cortex neural discharge to kinematics of passive and active elbow movements in the monkey," *J. Neurophysiol.*, vol. 60, no. 4, pp. 1268–1284, 1988.
[4] Y. Gao, M. J. Black, E. Bienenstock, S. Shoham, and J. P. Donoghue, "Probabilistic inference of hand motion from neural activity in motor cortex," in *Advances in Neural Information Processing Systems 14*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds. Cambridge, MA: MIT Press, 2002, pp. 213–220.
[5] Y. Gao, M. J. Black, E. Bienenstock, W. Wu, and J. P. Donoghue, "A quantitative comparison of linear and nonlinear models of motor cortical activity for the encoding and decoding of arm motions," in *Proc. 1st Int. IEEE/EMBS Conf. Neural Engineering*, Capri, Italy, Mar. 2003, pp. 189–192.
[6] A. Gelb, *Applied Optimal Estimation*. Cambridge, MA: MIT Press, 1974.
[7] A. Georgopoulos, J. Kalaska, R. Caminiti, and J. Massey, "On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex," *J. Neurosci.*, vol. 8, no. 11, pp. 1527–1537, 1982.
[8] A. Georgopoulos, R. Kettner, and A. Schwartz, "Primary motor cortex and free arm movements to visual targets in three-dimensional space. ii. Coding of the direction of movement by a neuronal population," *J. Neurosci.*, vol. 8, no. 8, pp. 2928–2939, 1988.
[9] A. Georgopoulos, A. Schwartz, and R. Kettner, "Neural population coding of movement direction," *Science*, vol. 233, pp. 1416–1419, 1986.
[10] Z. Ghahramani and G. E. Hinton, "Switching State-Space Models," Dept. Comp. Sci., Univ. Toronto, Toronto, ON, Canada, Tech. Rep. CRG-TR-96-3, 1996.
[11] A. Ghazanfar, C. Stambaugh, and M. Nicolelis, "Encoding of tactile stimulus location by somatosensory thalamocortical ensembles," *J. Neurosci.*, vol. 20, no. 10, pp. 3761–3775, 2000.
[12] P. L. Gribble and S. H. Scott, "Method for assessing directional characteristics of nonuniformly sampled neural activity," *J. Neurosci. Meth.*, vol. 113, no. 2, pp. 187–197, 2002.
[13] S. H.S. Helms Tillery, D. Taylor, R. Isaacs, and A. Schwartz, "Online control of a prosthetic arm from motor cortical signals," *Soc. Neurosci. Abst.*, vol. 26, 2000.
[14] Plexon Inc.. (2003), Dallas, TX. [Online] Available: http://www.plexoninc.com/ofs.htm.
[15] R. Kettner, A. Schwartz, and A. Georgopoulos, "Primary motor cortex and free arm movements to visual targets in three-dimensional space. iii: Positional gradients and population coding of movement direction from various movement origins," *J. Neurosci.*, vol. 8, no. 8, pp. 2938–2947, 1988.
[16] E. Maynard, C. Nordhausen, and R. Normann, "The Utah intracortical electrode array: a recording structure for potential brain-computer interfaces," *Electroencephalogr. Clin. Neurophysiol.*, vol. 102, pp. 228–239, 1997.
[17] D. Moran and A. Schwartz, "Motor cortical activity during drawing movements: population representation during spiral tracing," *J. Neurophysiol.*, vol. 82, no. 5, pp. 2693–2704, 1999.
[18] ——, "Motor cortical representation of speed and direction during reaching," *J. Neurophysiol.*, vol. 82, no. 5, pp. 2676–2692, 1999.
[19] K. P. Murphy, "Switching Kalman filter," Compaq Cambridge Res. Lab., Cambridge, MA, Tech. Rep. 98-10, 1998.
[20] M. A. L. Nicolelis, A. A. Ghazanfar, B. M. Faggin, S. Votaw, and L. Oliveira, "Reconstructing the engram: simultaneous, multisite, many single neuron recordings," *Neuron*, vol. 18, pp. 529–537, 1997.
[21] L. Paninski, M. Fellows, N. Hatsopoulos, and J. P. Donoghue, "Spatiotemporal tuning of motor cortical neurons for hand position and velocity," *J. Neurophysiol.*, vol. 91, pp. 515–532, 2004.
[22] J. Sanchez, D. Erdogmus, Y. Rao, J. Principe, M. Nicolelis, and J. Wessberg, "Learning the contributions of motor, premotor, and posterior parietal cortices for hand trajectory reconstruction in a brain machine interface," in *Proc. 1st Int. IEEE/EMBS Conf. Neural Engineering*, Caprí, Italy, Mar. 2004, pp. 59–62.
[23] A. Schwartz, "Motor cortical activity during drawing movements: population representation during sinusoid tracing," *J. Neurophysiol.*, vol. 70, no. 1, pp. 28–36, 1993.

[24] A. Schwartz and D. Moran, "Motor cortical activity during drawing movements: population representation during lemniscate tracing," *J. Neurophysiol.*, vol. 82, no. 5, pp. 2705–2718, 1999.
[25] A. Schwartz, D. Taylor, and S. H.S. Helms Tillery, "Extraction algorithms for cortical control of arm prothetics," *Curr. Opinion Neurobiol.*, vol. 11, pp. 701–707, 2001.
[26] M. Serruya, N. Hatsopoulos, and J. Donoghue, "Assignment of primate m1 cortical activity to robot arm position with bayesian reconstruction algorithm," *Soc. Neurosci. Abst.*, vol. 26, 2000.
[27] M. Serruya, N. Hatsopoulos, M. Fellows, L. Paninski, and J. Donoghue, "Robustness of neuroprosthetic decoding algorithms," *Biolog. Cybern.*, vol. 88, no. 3, pp. 201–209, 2003.
[28] M. D. Serruya, N. G. Hatsopoulos, L. Paninski, M. R. Fellows, and J. P. Donoghue, "Brain-machine interface: instant neural control of a movement signal," *Nature*, vol. 416, pp. 141–142, 2002.
[29] D. Taylor, S. H.S. Helms Tillery, and A. Schwartz, "Direct cortical control of 3D neuroprosthetic devices," *Science*, vol. 296, no. 5574, pp. 1829–1832, 2002.
[30] D. Wackerly, W. Mendenhall, and R. Scheaffer, *Mathematical Statistics With Applications*. Belmont, CA: Duxbury Press, 1996.
[31] D. Warland, P. Reinagel, and M. Meister, "Decoding visual information from a population of retinal ganglion cells," *J. Neurophysiol.*, vol. 78, no. 5, pp. 2336–2350, 1997.
[32] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," Univ. North Carolina at Chapel Hill, Tech. Rep. 95-041, 2001.
[33] J. Wessberg, C. Stambaugh, J. Kralik, M. Laubach, P. Beck, J. Chapin, J. Kim, S. Biggs, M. Srinivasan, and M. Nicolelis, "Real-time prediction of hand trajectory by ensembles of cortical neurons in primates," *Nature*, vol. 408, pp. 361–365, 2000.
[34] F. Wood, M. J. Black, C. Vargas-Irwin, M. Fellows, and J. P. Donoghue, "On the variability of manual spike sorting," *IEEE Trans. Biomed. Eng.*, vol. 51, pp. 912–918, June 2004.
[35] W. Wu, M. J. Black, Y. Gao, E. Bienenstock, M. Serruya, and J. P. Donoghue, "Inferring hand motion from multi-cell recordings in motor cortex using a kalman filter," in *SAB'02-Workshop on Motor Control in Humans and Robots: On the Interplay of Real Brains and Artificial Devices*, Edinburgh, U.K., Aug. 2002, pp. 66–73.
[36] W. Wu, M. J. Black, Y. Gao, E. Bienenstock, M. Serruya, A. Shaikhouni, and J. P. Donoghue, "Neural decoding of cursor motion using a Kalman filter," in *Advances in Neural Information Processing Systems 15*. Cambridge, MA: MIT Press, 2003, pp. 133–140.



Wei Wu was born in GuiYang, China, in 1974. He received the B.S. degree in applied mathematics (in 1998) from the University of Science and Technology of China and the M.S. degrees in applied mathematics (in 2000) and computer science (in 2003) from Brown University, Providence, RI, where he is currently working toward the Ph.D. degree in the Division of Applied Mathematics.

His research interests are in statistical models of neural coding and their applications to brain-machine interfaces.

Michael J. Black (S'90–M'92) received the B.Sc. degree from the University of British Columbia, Vancouver, BC, Canada, in 1985, the M.S. degree from Stanford University, Stanford, CA, in 1989, and the Ph.D. degree in computer science from Yale University,New Haven, CT, in 1992.

He has been a Visiting Researcher at the NASA Ames Research Center, Moffett Field, CA, and an Assistant Professor in the Department of Computer Science, University of Toronto, Toronto, ON, Canada. In 1993, he joined the Xerox Palo Alto Research Center where he managed the Image Understanding area and later founded the Digital Video Analysis Group. In 2000, he joined the faculty of Brown University, where he is currently an Associate Professor of Computer Science. His research explores probabilistic and statistical methods for optical flow estimation, human motion analysis, computational neuroscience, and neural prostheses.

Dr. Black received the IEEE Computer Society Outstanding Paper Award in 1992 for his work (with P. Anandan) on robust optical flow estimation. His work on the probabilistic detection and tracking of motion discontinuities (with D. Fleet) received Honorable Mention for the David Marr Prize (ICCV'99).

David Mumford (M'85) received the Ph.D. degree in mathematics from Harvard University, Cambridge, MA, in 1961.

He was a Professor at Harvard University until 1996, and subsequently has been a University Professor of Applied Mathematics at Brown University, Providence, RI. He worked in algebraic geometry from 1958 until 1983. Since then, his interests have been in the applications of mathematics and statistics to the modeling of thought, both computationally and in the brain, and especially in vision.

Dr. Mumford was President of the International Mathematical Union from 1994–1998 and is a member of the AMS and SIAM.

Yun Gao was born in Wuhan, China, in 1973. She received the B.S. degree in mathematics from the University of Science and Technology of China in 1997, the M.S. degree in applied mathematics in 2001 and the M.S. degree in computer science in the 2002, both from Brown University, Providence, RI, where she is currently working toward the Ph.D. degree in the Division of Applied Mathematics.

Elie Bienenstock received the Ph.D. degree in applied mathematics from Brown University, Providence, RI, in 1980.

He was a Research Fellow with CNRS, Paris, France, until 1990, and is now an Associate Professor of Applied Mathematics and Neuroscience at Brown University. His research interests in computational neuroscience include synaptic plasticity and fine temporal coding. His work in computer vision focuses on the study of Bayesian compositional models for image interpretation. He has contributed to the development of algorithms for optical character recognition.

John P. Donoghue (M'03) was born in Cambridge, MA, in 1949. He received the Ph.D. from Brown University, Providence, RI, in 1979.

He is a Professor and Chair of the Department of Neuroscience at Brown University. He has helped develop Brown's undergraduate neuroscience concentration and the brain science program, which brings together ten departments and more than 100 faculty into a unique interdisciplinary research and education program. His personal research program is aimed at understanding how the brain turns thought into movement and the development of neuromotor prosthetic devices. In addition, he is a Co-Founder of Cyberkinetics, Inc., Foxboro, MA, a biotech startup that is developing brain implants to restore movements to paralyzed individuals.